

Reductions

+

Turing

Machine

Variants

Properties of Reductions

$$\boxed{A \leq_m B} \Downarrow$$

$\exists$ total computable

$$\boxed{f: \Sigma^* \rightarrow \Sigma^* \text{ p.t.}}$$

$$\underline{\forall x \in \Sigma^*}$$

$$\underline{x \in A} \iff \underline{f(x) \in B}$$

Prop 1. If $A \leq_m B$, A is not recog.
then B is not recog.

Proof

Assume B is recog. by machine M_B

f is computable M

M_A : on input x :

1) Compute $f(x)$

2) Run M_B on $f(x)$

2.1) accept if accept, rej if rej.

$$x \in A \iff f(x) \in B$$

$$x \in A$$

$$\Downarrow$$

$$f(x) \in B$$

$\Downarrow$
 M_B will acc. $f(x)$

$\Downarrow$
 M_A will acc. x

$$x \notin A$$

$$\Downarrow$$

$$f(x) \notin B$$

$\Downarrow$
 M_B will loop/reject $f(x)$

$\Downarrow$
 M_A will loop/reject x

Prop 2. If $A \leq_m B$, then $\overline{A} \leq_m \overline{B}$

Proof:

comp. f
func.

$$x \in A \iff f(x) \in B$$

$$\underbrace{x \notin A} \iff \underbrace{f(x) \notin B}$$

$\Downarrow$

f is a reduction $\overline{A} \leq_m \overline{B}$

Prop 3. If $A \leq_m B$ & $B \leq_m C$, then
 $A \leq_m C$

Proof. Transitivity

$$A \leq_m B \quad B \leq_m C$$

$$\begin{array}{c} \underline{f} \\ M_1 \end{array}$$

$$\begin{array}{c} \underline{g} \\ M_2 \end{array}$$

$$\underline{A \leq_m C}$$

$$\underline{x \in A \Leftrightarrow h(x) \in C}$$

$R:$

$M_3:$ on input x :

- 1) computes $f(x)$ using M_1 ,
- 2) computes $\underline{g(f(x))}$ using M_2
- 3) Output $\underline{g(f(x))}$

$$\underline{x \in A \Leftrightarrow f(x) \in B \Leftrightarrow g(f(x)) \in C}$$

TR, coTR, decidable, none

$$\textcircled{1} \quad \boxed{E_{TM}} = \{x : L(M_x) = \emptyset\}$$

$$\bar{E}_{TM} = \{x : L(M_x) \neq \emptyset\}$$

recog.
Input M

for $i := 1$ to ∞

for $x \in \Sigma^*[1:i]$

Run M on x for i steps

Accept if acc.

$$\bar{A}_{TM} \leq_m E_{TM}$$

$$\langle M, w \rangle \mapsto M' = f(\langle M, w \rangle)$$

M' on input w :

1) Run M on w

2) reject if reject
acc. if accept

$\epsilon, 0, 1, 00, 01, 10, 11, \dots$

$$\Sigma^* \quad M, w \in \bar{A}_{TM}$$

M loops on w

M rejects w

M' loops on everything

M' rejects w

$$L(M') = \emptyset$$

$$L(M') = \emptyset$$

$$M, w \in \bar{A}_{TM} \Rightarrow M \text{ acc. } w$$

$$M' \text{ acc all things} \Rightarrow L(M') = \Sigma^* \neq \emptyset$$

$$M' \notin E_{TM}$$

$$\textcircled{2} \quad \underline{ALL_{TM}} = \{x : L(\underline{M_x}) = \Sigma^*\}$$

$$\overline{ALL_{TM}} = \{x : L(M_x) \neq \Sigma^*\}$$

$$A(M_x)$$

$$\textcircled{2} \quad ALL_{TM} = \{x : L(M_x) = \Sigma^*\}$$

$$\overline{ALL}_{TM} = \{x : L(M_x) \neq \Sigma^*\}$$

$$\underline{A}_{TM} \leq_m \underline{ALL}_{TM}$$

$$\underline{E}_{TM} \leq_m ALL_{TM}$$

$f(M) = M'$: on input x :

- 1) Run M on w
- 2) Acc. if accept

$f(M) = M'$: on input (x, t) :

- 1) Run M on x for t steps
- 2) If acc. in t steps, reject.
- 3) else accept

$$(M, w) \notin A_{TM}$$

$$(M, w) \notin A_{TM}$$

reject or loop

$$L(M') = \Sigma^*$$

$$M' \in ALL_{TM}$$

$$L(M') = \emptyset$$

$$M' \notin ALL_{TM}$$

$$M \in E_{TM}$$

$$\Downarrow$$

$$L(M) = \emptyset$$

$$\Downarrow$$

$$\forall x \quad M \text{ not acc. } x$$

$$\Downarrow$$

$$\underline{M'} \text{ accepts all } (x, t)$$

$$\Downarrow$$

$$L(M') = \Sigma^*$$

$$\underline{M \notin E_{TM}}$$

$$\Downarrow$$

$$\exists x, \text{ s.t. } M \text{ acc. } x \text{ after } t \text{ steps}$$

$$\Downarrow$$

$$M' \text{ rejects } (x, t)$$

$$\Downarrow$$

$$L(M') \neq \Sigma^*$$

$$\textcircled{3} \quad \underline{EQ}_{TM} = \{ \langle x, y \rangle : L(\underline{M_x}) = L(\underline{M_y}) \}$$

$$\overline{EQ}_{TM} = \{ \langle x, y \rangle : L(M_x) \neq L(M_y) \}$$

$$\textcircled{3} E Q_{TM} = \{ \langle x, y \rangle : L(M_x) = L(M_y) \}$$

$$\overline{EQ}_{TM} = \{ \langle x, y \rangle : L(M_x) \neq L(M_y) \}$$

M, w

$$\underline{A_{TM}} \leq_m E Q_{TM}$$

M_1 : on input x :

- 1) Run M on w
- 2) accept if acc.

M_2 : on input x :

- 1) accept

$$f(\langle M, w \rangle) = (M_1, M_2)$$

$$\begin{aligned} (M, w) \in A_{TM} & \quad L(M_2) = \Sigma^* \\ \Downarrow & \\ L(M_1) = \Sigma^* & = \end{aligned}$$

$$(M, w) \notin A_{TM}$$

$$\begin{aligned} \Downarrow & \\ L(M_1) = \emptyset & \neq \Sigma^* = L(M_2) \end{aligned}$$

$$\underline{\overline{A_{TM}}} \leq E Q_{TM}$$

M_1 : on input x :

- 1) Run M on w
- 2) accept if acc.

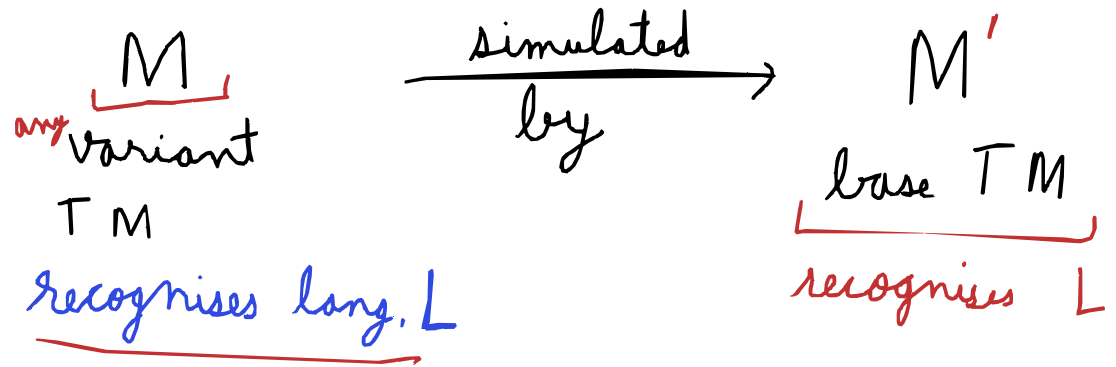
M_2 : on input x :

- 1) reject

practice

Turing Machine Variants

⇒ Simulate:



Sipser

Sec 3.2

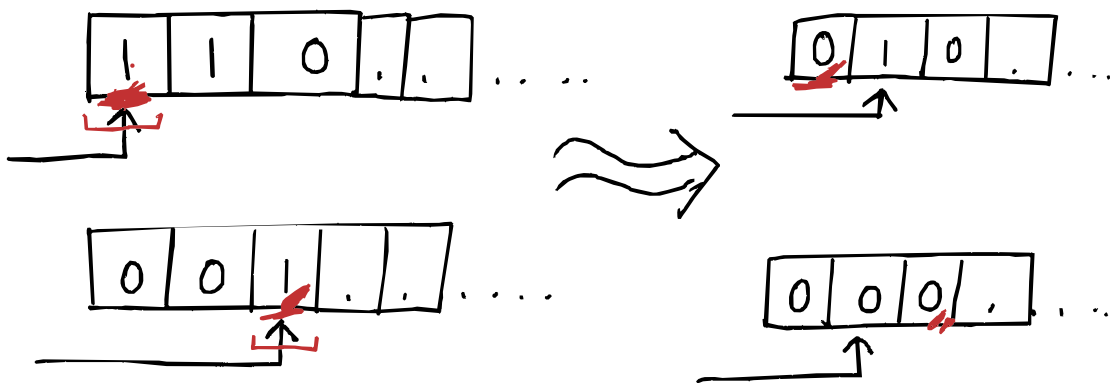
1) Multitape TMs.

⇒ Each tape has a head that moves independently.

Transition func.

$$\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R\}^k$$

eg. Transition



$$\delta(q_1, (1, 1)) \rightarrow (q_2, (0, 0), (R, L))$$

Thm. Every k -tape TM
can be simulated by a
base TM.

Proof:

We will show a two-tape
TM can be simulated by a 1-tape
TM

Thm. Every k -tape TM
can be simulated by a
base TM.

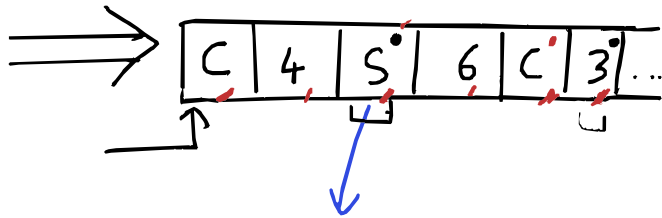
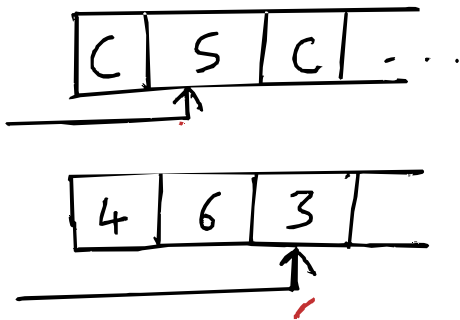
Proof:

We will show a two-tape
TM can be simulated by a 1-tape
TM

Idea:

- Odd tape cells encode tape 1
- Even tape cells encode tape 2.
- Use dotted characters for head position.

eg.



S' new symbol denoting head is here.

Q

Γ

δ

q_0

F



$Q \times \{1, 2, 3, 4\}$

$\Gamma \cup \Gamma'$

δ'

$(q_0; 1)$

F

For k

use $\begin{cases} 1, k+1, 2k+1, \dots & \text{for tape 1} \\ 2, k+2, 2k+2, \dots & \text{for tape 2} \\ \vdots & \end{cases}$

Simulate transition

$$\delta(q, (a, b)) \rightarrow (q', \underline{(c, d)}, \{L/R, L/R\})$$

Step 1: Scan tape to find value of tapes at each tape head $\underline{(a, b)}$

Step 2: Transition to new state depending on $\underline{\delta}$ $\leftarrow$ write $\underline{(c, d)}$
 $\underline{q'}$

Step 3: Another scan to replace values and change tape heads (the $\underline{\bullet}$)

Look in book
on how to
write Pg 149

2. Non-Deterministic TMs.

⇒ use Nondeterminism same as NFA.

⇒ At any point, TM may make a choice out of many possible on what to do.

⇒ We say TM accepts an input if some seq. of choices cause TM to accept.

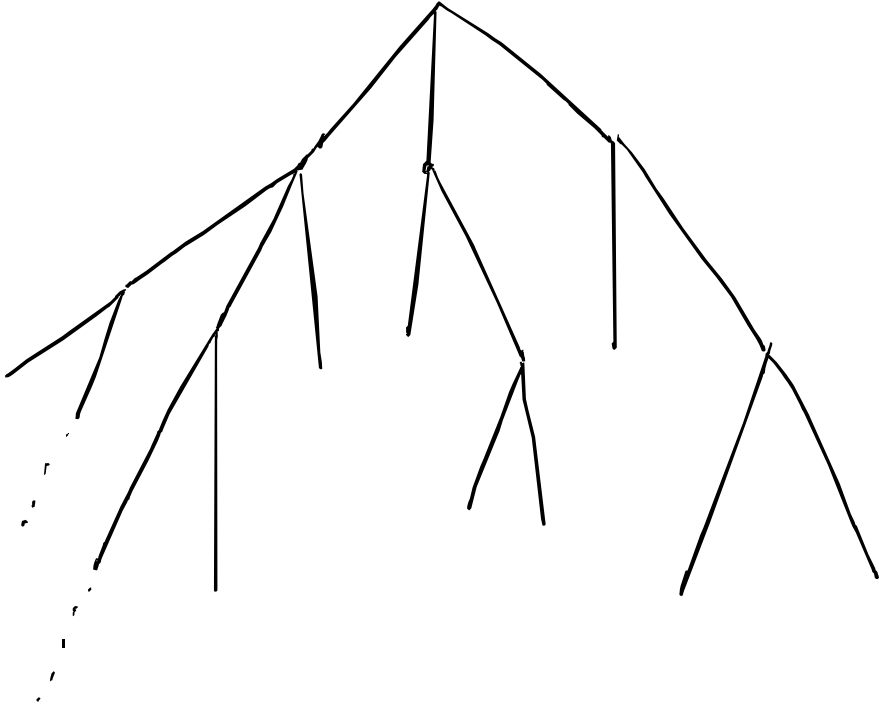
$$|Q| \times |T| \times 2$$

⇒ Transition func.

$$\delta: \underline{Q} \times \underline{T} : \underbrace{\mathcal{P}}_{\text{Power set}} \left(\underline{Q} \times \underline{T} \times \underline{\{L, R\}} \right)$$

⇒ Tran. func. gives a set of instruc. to choose from.

Eg. How NDTM computation looks like



any accept means NDTM accepts.

Thm. Every NDTM can
be simulated by a base
TM.

Thm. Every NDTM can
be simulated by a base
TM.

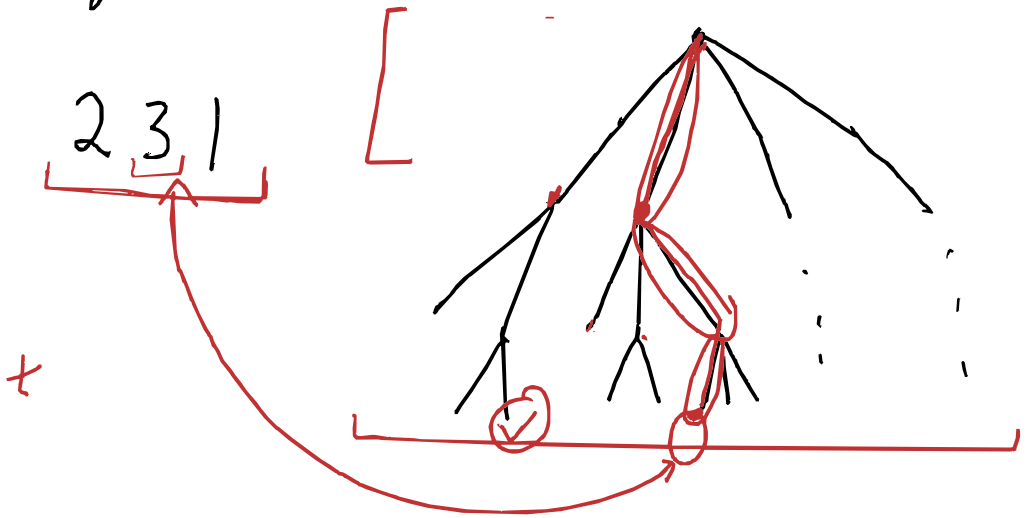
Idea: do BFS of the tree

⇒ use 3-tape TM

⇒ At each node, at most
choices. $b = |Q \times \Gamma \times \{L, R\}|$

⇒ Address of node at level k
is a string in $\{1, 2, \dots, b\}^k$

eg.



$\Rightarrow$ BFS $\longleftrightarrow$ searching the
addresses in SSO
 $\{1, 2, \dots, b\}^*$

$\{ \underline{\epsilon}, 1, 2, \dots, b, 11, \underline{12}, \dots \}$

$\Rightarrow$ tape 1 : input

tape 2 : used for simulation

tape 3 : address

⇒ Base Machine M simulating NDTM
N
↓

while True:

→ Copy tape 1 to tape 2

→ Run N on tape 2, whenever a non-det. choice is to be made, consult tape 3.

→ If found node is accepting:
accept.

→ increment tape 3, clear tape 2.